Albahit Journal of Applied Sciences
"open-access, peer-reviewed biannual journal"
eISSN: 2708-8936, pISSN: 2708-244X
Volume 5, Issue 3, 2026
Page No: 146-163
***Website**: https://albahitjas.com.ly/index.php/albahit/en/index*

Chattering Under Load: Control-Theoretic and Trajectory-Level Behavior of Lion and AdamW Under Physical Perturbation in PPO

Osama Meftah Wanis [1*], Hamza Alzarok [2]
[1] College of Electronic Technology, Bani-Walid, Libya
[2] Department of Electrical and Electronic Engineering, Faculty of Engineering, Bani Waleed University, Bani Waleed, Libya
[*] *Corresponding author:* osamayagaa@gmail.com

Received: 17-05-2026	**Accepted:** 22-07-2026	**Published:** 05-08-2026

Abstract:
PPO policies rely in their training almost exclusively on AdamW, despite what the sign-based alternative Lion offers in reducing the optimizer-state memory footprint by half. However, this structural advantage is paired with a decisive structural cost: Lion's update magnitude is determined directly and absolutely by the instantaneous learning rate, entirely stripped of any self-normalizing compensation mechanism such as the one AdamW possesses. This structural deficiency reduces the update process to something equivalent to a relay/bang-bang controller behavior applied within the parameter space, which makes it inevitably prone to the chattering phenomenon — the persistent high-frequency oscillations that classical theoretical frameworks of Sliding Mode Control and Variable-Structure Control attribute to discontinuous, sign-based control laws. This paper adopts an analytical perspective derived from control theory to investigate this behavior under the weight of physical perturbation across five continuous-control MuJoCo environments — HalfCheetah-v4, Walker2d-v4, Ant-v4, and Hopper-v4, in addition to the extended-horizon Humanoid-v4 — relying on precise joint-level diagnostics for each environment (contact force, actuator saturation, torque chattering, policy entropy), and supported by an integrated physical-perturbation robustness protocol (adding mass by +20% and +50%, and sensor noise, followed by a short fine-tuning re-adaptation phase), at a strict statistical depth of n = 15 seeds. The outputs of this research are organized around three central findings. First, sensor noise — unlike added mass — causes a confirmed statistically significant increase (Welch's t-test, Holm–Bonferroni corrected) in chattering for most arms across the four main environments (for example, Hopper-v4 lion_cosine: d = −3.50; and Walker2d-v4 lion_warmup_cosine: d = −3.42). This behavior has an exact counterpart in classical control, where measurement noise near the sliding surface is considered the primary trigger for chattering in the sliding-mode control literature. Second, the pattern of post-fine-tune return recovery varies radically depending on the arm within a single environment, and not only depending on the environment: in the Walker2d-v4 environment, lion_cosine recovers only 17–53% of its own clean-condition return depending on the type of perturbation, the lowest recovery profile for any arm in this study, despite being among the most efficient in clean conditions within that environment. Third, the exploratory extension to the Humanoid-v4 environment (after 5000 iterations, roughly 10.24M steps, with all arms and conditions ending in falls) reveals AdamW's superiority in retaining a higher fraction of its clean-condition return under the three perturbations compared with any Lion variant — a clear reversal and

contradiction of the results recorded in clean conditions there. We present the representative state trajectories for the Humanoid-v4 environment and connect their aggregate patterns to a sliding-mode-control-based interpretation of the chattering phenomenon, along with explicit methodological caveats where episode-length truncation causes confounding and distortion of control-quality metrics.

Keywords: Proximal Policy Optimization, Lion optimizer, AdamW, sliding-mode control, chattering, robustness to physical perturbation, MuJoCo.

التذبذب عالي التردد تحت الحمل: السلوك النظري-التحكمي وعلى مستوى المسارات لمُحسِّنَي *Lion* و*AdamW* تحت الاضطراب الفيزيائي في *PPO*

أسامة مفتاح ونيس[1]، حمزة الزروق[2]
[1] كلية التقنية الإلكترونية، بني وليد، ليبيا
[2] قسم الهندسة الكهربائية والإلكترونية، كلية الهندسة، جامعة بني وليد، بني وليد، ليبيا

الملخص
تعتمد سياسات **PPO**في تدريبها بشكل شبه حصري على **AdamW**، على الرغم من أن البديل القائم على الإشارة **Lion** يوفر ميزة تقليل البصمة الذاكرية لحالة المُحسِّن إلى النصف. ومع ذلك، فإن هذه الميزة البنيوية تقترن بتكلفة بنيوية حاسمة: إذ إن مقدار التحديث في **Lion**يتحدد بشكل مباشر ومطلق بواسطة معدل التعلم اللحظي، مع تجريده بالكامل من أي آلية تعويض ذاتي للتطبيع مثل تلك التي يمتلكها **AdamW**. هذا القصور البنيوي يحول عملية التحديث إلى سلوك مكافئ لسلوك متحكم مرحلي/**ثنائي الحالة (Relay/Bang-Bang Controller)** مطبق داخل فضاء المعلمات، مما يجعله عرضة حتميًا لظاهرة **التذبذب عالي التردد(Chattering)** ، وهي التذبذبات المستمرة عالية التردد التي تربطها الأطر النظرية الكلاسيكية لكل من **التحكم بالانزلاق (Sliding Mode Control)** و**التحكم متغير البنية (Variable-Structure Control)**بقوانين التحكم غير المستمرة القائمة على الإشارة.
تتبنى هذه الورقة منظورًا تحليليًا مشتقًا من نظرية التحكم لدراسة هذا السلوك تحت تأثير الاضطرابات الفيزيائية عبر خمس بيئات للتحكم المستمر في — **MuJoCo**وهي **HalfCheetah-v4**، **Walker2d-v4**، **Ant-v4**، و**Hopper-v4**، بالإضافة إلى البيئة ذات الأفق الزمني الممتد — **Humanoid-v4**بالاعتماد على تشخيصات دقيقة على مستوى المفاصل لكل بيئة) **قوة التلامس، تشبع المشغلات، تذبذب العزم، وإنتروبيا السياسة**(، ومدعومة ببروتوكول متكامل لاختبار المتانة ضد الاضطرابات الفيزيائية (إضافة كتلة بنسبة **20%+و+50%**، وضوضاء المستشعرات، يليها مرحلة قصيرة من إعادة التكيف بواسطة الضبط الدقيق)، وذلك بعمق إحصائي صارم يبلغ **15 = n بذرة تجريبية**.
تُنظم مخرجات هذا البحث حول ثلاث نتائج رئيسية. أولًا، تؤدي ضوضاء المستشعرات — بخلاف إضافة الكتلة — إلى زيادة مؤكدة ذات دلالة إحصائية) **اختبار Welch's t-test مع تصحيح (Holm–Bonferroni** في **التذبذب عالي التردد** لدى معظم المُحسِّنات عبر البيئات الأربع الرئيسية (على سبيل المثال، **Hopper-v4 lion_cosine: d = −3.50**؛ و **Walker2d-v4 lion_warmup_cosine: d = −3.42**). يمتلك هذا السلوك نظيرًا مطابقًا في نظرية التحكم التقليدية، حيث تُعتبر ضوضاء القياس بالقرب من سطح الانزلاق العامل الأساسي المحفز لظاهرة التذبذب عالي التردد في أدبيات التحكم بالانزلاق.
ثانيًا، يختلف نمط استعادة العائد بعد الضبط الدقيق بشكل جذري اعتمادًا على المُحسِّن داخل البيئة الواحدة، وليس فقط اعتمادًا على اختلاف البيئة؛ ففي بيئة **Walker2d-v4** يستعيد **lion_cosine**نسبة تتراوح بين **53%–17**فقط من عائده الخاص في الحالة النظيفة، وذلك حسب نوع الاضطراب، وهو أقل نمط استعادة لأي مُحسِّن في هذه الدراسة، على الرغم من كونه من بين الأكثر كفاءة في الظروف النظيفة داخل تلك البيئة.
ثالثًا، يكشف الامتداد الاستكشافي إلى بيئة) **Humanoid-v4**بعد **5000دورة تدريبية**، أي ما يقارب **10.24مليون خطوة**، مع انتهاء جميع المُحسِّنات وجميع الظروف بحالات سقوط) عن تفوق **AdamW**في الاحتفاظ بنسبة أعلى من عائده في الظروف النظيفة تحت الاضطرابات الثلاثة مقارنة بأي نسخة من نسخ **Lion**، وهو انعكاس واضح وتناقض مع النتائج المسجلة في الظروف النظيفة هناك. نقدم المسارات التمثيلية للحالات لبيئة **Humanoid-v4**، ونربط الأنماط التجميعية

الخاصة بها بتفسير قائم على التحكم بالانزلاق لظاهرة **التذبذب عالي التردد**، إلى جانب عرض القيود المنهجية بشكل صريح في الحالات التي يؤدي فيها تقصير طول الحلقة (**Episode-Length Truncation**) إلى حدوث عوامل مربكة وتشويه لمقاييس جودة التحكم.

الكلمات المفتاحية: التحسين القريب من السياسة، مُحسِّن **Lion**، **AdamW**، التحكم بالانزلاق، **التذبذب عالي التردد** ، المتانة ضد الاضطرابات الفيزيائية.

Introduction

The Lion optimizer (Evolved Sign Momentum; [1]) was introduced via symbolic program search, achieving a reduction of half the optimizer state memory compared with AdamW, owing to its reliance on a single momentum buffer and its application of a fixed-magnitude coordinate-wise sign update instead of AdamW's adaptive, dual-buffer magnitude update. Sign-based updates are not exclusive to Lion; prior research such as SignSGD [2] has demonstrated that coordinate-wise sign updates are capable of preserving competitive convergence while reducing communication and storage costs, although that is separate from the control-theory analytical framework this paper adopts. This magnitude is computed entirely via the instantaneous learning rate, stripped of any mechanism resembling AdamW's self-normalizing compensation (see the Method section for the detailed derivation); hence, a decaying schedule shrinks Lion's entire update capacity all at once rather than reducing it partially and in a balanced manner. This design, resting entirely on sign and fixed magnitude, is the direct cause of Lion falling under the sway of chattering: the persistent high-frequency oscillation historically documented in sliding-mode and variable-structure control for discontinuous, sign-based control laws [3]. We consider chattering here to be a literal technical property and not merely a figurative metaphor: Lion's update represents, geometrically and structurally, a relay or bang-bang controller operating in the parameter space. The Related Work discussion below situates this analytical framework against research on memory-/sign-efficient optimizers and research on sim-to-real robustness and physical perturbations in robotics.

This paper raises a question about how this control-theory-derived interpretation manifests under physical perturbations, at full seed depth, across all targeted environments, using per-environment diagnostic tables (contact force, actuator torque chattering, saturation, entropy) and supported by an integrated perturbation-and-fine-tuning protocol (pre/post fine-tune return, pre/post chattering and control energy, recovery time, Welch/Holm tests per arm and condition), all at a statistical rigor of $n = 15$ seeds, to isolate configuration properties from environment properties, and to test whether sliding-mode control theory possesses predictive power in determining the type of perturbation that triggers the phenomenon. We expanded the evaluation scope to include Humanoid-v4, the highest-dimensional environment in this study (17 actuated joints), which was trained for 5000 PPO iterations (roughly 10.24M steps, i.e., 3.3 to 5× the time horizon of the four primary environments). Here, we tracked the representative state trajectories, and compiled a complete battery of control-quality metrics under clean vs. perturbed conditions.

Methodological integrity requires establishing a pivotal fact regarding Humanoid-v4's performance before interpreting its figures alongside the four primary environments: **all policies evaluated on Humanoid-v4, under every arm and condition, including clean conditions, fall before the evaluation episode reaches the environment's nominal horizon** (see the Humanoid-v4 subsection below). Consequently, the metrics recorded there describe the dynamics of the fall itself, not a mature standing or walking controller, and we state this scope limitation to prevent any mistaken conclusion about stability that control-theory terminology might suggest. This caveat does not apply to the four primary environments, where fall rates differ by arm and many of them achieve sustained stability across the full time horizon.

Related Work

Lion's update, based exclusively on sign and fixed magnitude, falls under a broader direction aimed at reducing optimizer state memory, alongside SignSGD's per-coordinate sign compression [2], 8-bit block-wise quantized optimizer states [4], and low-rank gradient projection [5]. Lion's convergence rate has also been separately analyzed under the gradient ℓ_1 norm criterion [6], and cautious variants have since been proposed that nullify the update when its sign conflicts with the raw gradient [7]. However, none of this optimizer-focused literature evaluates behavior under a physical-perturbation protocol such as the one we adopt here; their validation is limited to static-target supervised objectives, rather than the self-generated, non-stationary data distribution that a PPO policy produces, nor does any of it examine trajectory-level control-quality indicators (contact force, torque chattering, actuator saturation) as a function of optimizer choice.

On another front, an independent line of legged-robotics research achieves robustness against perturbations and sim-to-real transfer through domain randomization and massively parallel training [8], instantaneous adaptation modules that infer changes in terrain or payload during operation [9,10], perceptive locomotion policies robust against irregular external terrain [11], and sim-to-field alignment for whole-body humanoid robot skills [12]. In all these cases, robustness is engineered via the training curriculum, the observation pipeline, or an explicit adaptation module, while the optimizer is treated as a fixed implementation component rather than as an independent variable; to our knowledge, the question of whether the optimizer/schedule choice itself modulates perturbation robustness — the question this paper isolates — has not been examined in this literature.

Finally, the chattering phenomenon tracked in this paper is derived directly from sliding-mode and variable-structure control theory [3], which describes the persistent oscillation resulting from a discontinuous, sign-based control law operating near the switching surface. Applying this interpretation specifically to the trajectory-level behavior of a sign-based deep-learning optimizer under physical perturbation — rather than applying it only to loss-function behavior — is, to our knowledge, something not addressed by the optimizer-memory literature or the robustness literature cited above; we treat this as positioning the paper between these two scientific fields rather than as an isolated claim.

Material and methods

Optimizer Update Rules and Step-Size Coupling

AdamW [13] extends Adam [14] by pulling weight decay out of the gradient-based update term:

$\theta_t = \theta_{t-1} - \eta_t \left(\hat{m}_t / (\sqrt{\hat{v}_t} + \delta) + \lambda\theta_{t-1} \right)$ (1)

Lion [1] works differently: it combines a momentum-blended direction signal $c_t = \beta_1 m_{t-1} + (1-\beta_1)g_t$ with a separately smoothed buffer $m_t = \beta_2 m_{t-1} + (1-\beta_2)g_t$, then steps only in the sign of c_t: $\theta_t = \theta_{t-1} - \eta_t(sign(c_t) + \lambda\theta_{t-1})$. This means every nonzero Lion coordinate update lands at exactly η_t, regardless of the underlying gradient's scale:

$|\Delta\theta_t| = \eta_t \text{ whenever } c_t \neq 0$ (2)

in contrast to AdamW's $|\Delta\theta_t| = \eta_t \cdot f(\hat{m}_t, \hat{v}_t)$, whose self-normalizing multiplier absorbs part of η_t's decay. Lion offers no such cushion: as a schedule decays, its step shrinks one-to-one with η_t, while AdamW's effective step remains partly insulated from the schedule. This structural asymmetry is why we track *chattering*, $C_{chatter}$ — the mean absolute first difference of the evaluation action trace, $|a_t - a_{t-1}|$, averaged first within an episode and then across episodes and seeds within an arm — as a control-smoothness outcome throughout this paper.

A Structural Justification of Chattering: Genuine Non-Smoothness, Not Metaphor

The relay/bang-bang analogy above is not merely descriptive: it corresponds to a genuine non-smoothness in the PPO-clip objective as a function of θ once θ departs from θ_{old}, since the clipping boundary $r_t(\theta) = 1 \pm \varepsilon$ is then generically active.

At the single point $\theta = \theta_{old}$, the importance ratio $r_t(\theta)$ is identically 1, so the clip is never active there and the per-sample objective reduces to the smooth policy-gradient identity $g_t(\theta_{old}) = -\hat{A}_t \nabla_\theta \log \pi_\theta(a_t|s_t)$; this is a single-point fact, not a property of the algorithm's actual trajectory. In every PPO implementation used here, $K_{epoch} = 10$ inner Lion steps are taken per rollout (Table 1), so the inner iterate θ moves away from θ_{old} almost immediately, and the clip boundary becomes generically active. The relevant regularity result therefore has to hold on the whole traversed region, not just at one point: writing $\ell_t(\theta) := -f_t(\theta; \theta_{old})$, the max/min decomposition of the clipped objective shows that ℓ_t is a pointwise maximum (or minimum) of one smooth branch and one constant, and is therefore Clarke-regular [15] on the whole parameter space Θ for every sample *t*, not only in a neighborhood of θ_{old}. Because a full outer PPO update runs $M = K_{epoch} \times \lceil T/|B| \rceil = 320$ Lion steps on a single frozen minibatch-sampled dataset before the rollout is refreshed, the inner iterate genuinely traverses this non-smooth region at every outer step, not just occasionally — this is the precise sense in which the relay/switching-surface analogy above is a structural fact about the optimization landscape rather than a post-hoc description of its output. A schedule-dependent convergence analysis quantifying how this non-smoothness interacts with the optimizer's step-size schedule is developed as part of a companion empirical study of schedule effects and is outside the present paper's scope, which isolates the perturbation-robustness question instead.

Environments, Arms, and Statistical Framework

The comparison arms in this study consist of five specific models — namely adamw, adamw_tuned, lion_raw, lion_cosine, and lion_warmup_cosine, each represented at a statistical depth of $n = 15$ seeds — following their training via the PPO algorithm [16] across four standard MuJoCo [17] environments available in the Gymnasium library [18]: namely HalfCheetah-v4, Walker2d-v4, Ant-v4, and Hopper-v4. These environments cover a graduated spectrum of kinematic sensitivity, ranging from passively stable systems such as HalfCheetah-v4 and Ant-v4, up to Hopper-v4, which is prone to tipping over and falling and is characterized by extreme balance sensitivity. The adamw model is organized within the original baseline configuration of the PPO algorithm [16]; while adamw_tuned adopts the hyperparameters adopted in Stable-Baselines3's RL Zoo [19] alongside a mismatched rollout length — a discrepancy we consider to compromise direct comparability given the high sensitivity that PPO exhibits toward precisely this type of implementation detail [20,21] — and hence we limit its presentation to purely descriptive analysis, with its complete exclusion from the correlated inferential tests subject to Holm statistical correction. The equivalent-condition models (lion_raw, lion_cosine, and lion_warmup_cosine) match completely with adamw's network architecture, GAE parameters [22], and rollout length ($T = 2048$), while the differentiation between them is limited to the adopted optimizer and the learning-rate schedule (constant rate; cosine decay starting from step 0; and cosine decay preceded by linear warmup and a plateau period). Furthermore, the four equivalent-condition models (excluding adamw_tuned due to the absence of its evaluation data in this environment) were subjected to training on the Humanoid-v4 environment — the highest-dimensional and most complex environment in this study, with 17 actuated joints — for 5000 PPO iterations (≈10.24M steps), as an exploratory extension applying the same protocol and seed depth.

The statistical tests adopted across all axes of the study are based on Welch's *t*-test in conjunction with the application of Holm–Bonferroni correction [23] across two separate statistical families: the first family is the six pairwise comparisons per environment among the four equivalent-condition models (excluding adamw_tuned, consistent with its descriptive role specified above); the second family is each model's metrics in its clean conditions versus each

perturbation condition, with the statistical correction applied per environment across all five models and all tested conditions for this metric — and it is specifically within this second family that adamw_tuned's internal shifts between clean and perturbed conditions are recorded, consistent with analyzing it descriptively rather than comparing it directly with the other models. Effect sizes computed for all models are represented by Cohen's *d* [24], following the statistical power analysis guidelines of [25] for seed-based comparisons in reinforcement learning; a negative value indicates that the balance tips in favor of the other party at the expense of the model mentioned first (or of clean conditions, for the second family).

Hyperparameters and Compute

Optimizer and PPO hyperparameters for the four matched-condition arms (adamw, lion_raw, lion_cosine, lion_warmup_cosine) are shared across all four primary environments and given in Table 1; adamw_tuned's hyperparameters instead follow Stable-Baselines3's RL Zoo configuration and vary by environment (learning rate, PPO epochs, minibatch size, clip range, GAE λ, and network width), consistent with its descriptive, non-matched-condition role above. Both policy and value networks are two-hidden-layer MLPs with Tanh activations and orthogonal initialization. Full per-environment values for adamw_tuned are given in the Appendix (Table 6).

Table 1. Shared PPO/optimizer hyperparameters, four matched-condition arms, all four primary environments. adamw_tuned's hyperparameters vary by environment and are given in the Appendix.

Hyperparameter	adamw	lion_raw	lion_cosine	lion_warmup_cosine
Initial learning rate η_0	3×10^{-4}	3×10^{-5}	3×10^{-5}	3×10^{-5}
LR schedule	constant	constant	cosine decay, no warmup	linear warmup → cosine decay
Terminal LR η_T	3×10^{-4}	3×10^{-5}	$\approx1\times10^{-6}$	$\approx1\times10^{-6}$
PPO epochs/update	10	10	10	10
Minibatch size	64	64	64	64
Rollout n_steps	2048	2048	2048	2048
β_1	0.9	0.9	0.9	0.9
β_2	0.999	0.99	0.99	0.99
Weight decay λ	0.0	0.01	0.01	0.01
Clip range ε	0.2	0.2	0.2	0.2
Discount γ	0.99	0.99	0.99	0.99
GAE λ	0.95	0.95	0.95	0.95
Hidden layer width	128/128	128/128	128/128	128/128
Random seeds	15	15	15	15

Training was run on Google Colab (free-tier quota) using the standard CPU runtime; no GPU or TPU accelerator was allocated for any of the experiments reported in this paper. Colab's free-tier CPU runtime provisions an Intel Xeon processor with 2 virtual CPUs (vCPUs, clocked at approximately 2.2 GHz) and approximately 13 GB of RAM.

Metrics

In addition to measuring return and the chattering metric defined as $C_{chatter}$ (which is procedurally formulated as the mean absolute first difference of the evaluation action trace, and is defined identically throughout every part of this research), we report: mean and peak contact force and spike count (representing the frequency of ground-contact events with sharp values in each episode); actuator saturation fraction (representing the proportion of (evaluation

step, joint) cases in which the requested actions exceed 98% of the torque limit); and post-training policy entropy. Within the perturbation protocol, we additionally report: return before/after fine-tuning (measured as a percentage of the same model's clean return prior to applying the perturbation), chattering and control energy before/after fine-tuning, and recovery time (the total time elapsed within the fine-tuning window before the return settles again). The perturbation battery consists of: added mass (+20% and +50% of the default body mass), and injected sensor/observation noise, each followed by up to 100 fine-tuning steps for re-adaptation starting from the perturbed checkpoint. Specifically for the Humanoid-v4 environment, we additionally report: overshoot percentage, settling time for the torso-height trajectory, control energy, and representative per-timestep state trajectories for torso height z and pitch angle θ for one seed per model; see the Humanoid-v4 subsection below for the scope-related caveats concerning these metrics.

Results and discussion

Cross-Environment Actuator and Contact-Force Behavior

Table 2. Clean-condition joint-level diagnostics, all four primary environments, all five arms, $n = 15$ seeds. Compute hardware (CPU/GPU/RAM) was not recorded in the underlying logs and is not reported here.

Environment	Arm	Mean contact force (N)	Peak contact force (N)	Spike count/seed	Torque chattering	Saturation fraction
HalfCheetah-v4	adamw	131.5	3918	159	0.656	0.475
	adamw_tuned	131.6	3971	200	0.782	0.556
	lion_raw	143.6	4206	191	0.749	0.415
	lion_cosine	142.6	4248	211	0.800	0.430
	lion_warmup_cosine	149.3	3949	208	0.777	0.396
Walker2d-v4	adamw	268.6	3640.6	32.4	0.496	0.562
	adamw_tuned	262.8	4079.6	46.0	0.354	0.438
	lion_raw	260.6	3912.9	33.1	0.360	0.414
	lion_cosine	268.7	3986.3	94.4	0.260	0.380
	lion_warmup_cosine	266.2	4092.8	71.0	0.291	0.418
Ant-v4	adamw	11.7	549	57	0.520	0.182
	adamw_tuned	12.6	613	125	0.215	0.024
	lion_raw	11.4	575	54	0.417	0.113
	lion_cosine	11.7	505	108	0.340	0.026
	lion_warmup_cosine	11.3	554	75	0.401	0.080
Hopper-v4	adamw	170.5	2730	93.5	0.317	0.312
	adamw_tuned	185.0	967	37.9	0.924	0.758
	lion_raw	163.9	2794	70.8	0.283	0.376
	lion_cosine	163.9	3040	82.1	0.207	0.411
	lion_warmup_cosine	165.8	3069	97.7	0.278	0.375

No single arm is uniformly high- or low-saturating, high- or low-contact-force, or high- or low-spike-count across all four environments (Table 2), indicating that morphology, not a fixed optimizer property, governs actuation demands. Two patterns are nonetheless consistent across at least three of the four environments.

- **Spike Count Escalation Under Cosine-Decay Schedule**: lion_cosine shows a higher spike count compared with lion_raw in the Walker2d-v4 environment (94.4 versus 33.1) and the Ant-v4 environment (108 versus 54), representing a widening of the gap by roughly 2 to 3-fold, with the same trend emerging, albeit with a narrow gap, in the Hopper-v4 environment (82.1 versus 70.8). This trajectory is consistent with the effect of decay on the paired torque-chattering column in Table 2 (where the value also registers an increase for lion_cosine compared with lion_raw across all three conditions).
- **Environment-Dependent Saturation Range for adamw_tuned**: the saturation fraction of adamw_tuned is the most environment-dependent among all five models; it peaks in the Hopper-v4 environment at 0.758 (roughly 1.8× the next-highest model), while it plunges to its lowest level in the Ant-v4 environment at 0.024 — a range that varies by a factor of 30, far exceeding by a wide margin the range of any equivalent-condition model across the different environments (for example, adamw: 0.182–0.562, a variation range not exceeding 3-fold).

Physical-Perturbation Return Recovery

Table 3. Post-fine-tune return as a percentage of each arm's own pre-perturbation clean return (Post-FT%), all four primary environments, all five arms, three perturbation conditions, n = 15 seeds/cell. Environment-level means, averaged across all five arms and three conditions: HalfCheetah-v4 88.4%, Walker2d-v4 54.7%, Hopper-v4 79.0%, Ant-v4 113.7%. Walker2d-v4 clean-condition mean returns (the denominators underlying that row block's Post-FT% values): adamw 782.4, lion_raw 740.7, lion_cosine 3541.8, lion_warmup_cosine 2153.9.

Environment	Arm	Mass +20%	Mass +50%	Sensor noise
HalfCheetah-v4	adamw	132.9%	110.7%	109.4%
	adamw_tuned	84.4%	75.9%	71.2%
	lion_raw	92.3%	63.2%	79.8%
	lion_cosine	95.8%	79.3%	90.4%
	lion_warmup_cosine	92.6%	68.3%	79.8%
Walker2d-v4	adamw	99.7%	68.1%	86.4%
	adamw_tuned	52.8%	45.9%	53.7%
	lion_raw	91.0%	58.1%	74.1%
	lion_cosine	**53.5%**	**17.0%**	**23.7%**
	lion_warmup_cosine	42.3%	21.8%	32.7%
Hopper-v4	adamw	125.8%	93.3%	41.5%
	adamw_tuned	99.8%	94.6%	58.8%
	lion_raw	121.8%	77.1%	39.8%
	lion_cosine	116.7%	95.6%	32.5%
	lion_warmup_cosine	96.0%	62.8%	29.2%
Ant-v4	adamw	128.1%	146.3%	96.5%
	adamw_tuned	94.1%	99.9%	61.3%
	lion_raw	**165.7%**	**166.6%**	**100.0%**
	lion_cosine	106.5%	122.5%	74.3%
	lion_warmup_cosine	126.4%	130.4%	87.2%

Table 3 reveals that arm-level variation within a single environment frequently equals, or exceeds, the environment-level variation summarized above.

- **Response Collapse in Walker2d-v4**: this environment records the lowest average recovery at the environment level (54.7%). lion_cosine — the highest-performing model by a clear margin in clean conditions within this environment (mean return of 3541.8, versus 782.4 for adamw, 740.7 for lion_raw, and 2153.9 for lion_warmup_cosine) — shows the worst recovery profile of any arm in the entire table (53.5%/17.0%/23.7%), while adamw recovers to a markedly higher fraction of 99.7%/68.1%/86.4%, despite the disadvantage of its much lower clean return in that environment.
- **Inverted Pattern in Ant-v4**: this environment records the highest average recovery at the environment level (113.7%). lion_raw — the weakest Lion model in clean conditions within this environment — shows the best recovery profile in the entire table (165.7%/166.6%/100.0%), achieving a clear jump relative to its clean baseline under both mass perturbations.

Recovery exceeding 100% indicates that the fine-tuned checkpoint surpasses its own pre-perturbation clean return, which more likely reflects an additional training signal derived from the fine-tuning window itself rather than being a substantive gain resulting from the perturbation; we record this behavior descriptively without attaching any inferential claims to it.

Recovery Time and Control Effort Before/After Fine-Tuning

Table 4. Recovery time (s) and chattering, before vs. after the fine-tuning re-adaptation window, mean ± SD across all three perturbation conditions and $n = 15$ seeds/arm. HalfCheetah-v4 has no fall-termination condition; its reported recovery time is effectively the full fine-tuning window duration (≈50 s) for every arm and is not a genuine convergence measure. Only two representative arms (adamw and lion_cosine) are shown for HalfCheetah-v4, as all five arms saturate to the same ≈50 s ceiling; the remaining three arms show the identical censored pattern and are omitted for brevity.

Environment	Arm	Recovery, before (s)	Recovery, after (s)	Chattering, before	Chattering, after
HalfCheetah-v4	adamw	49.993 ± 0.020	49.993 ± 0.011	0.653 ± 0.142	0.682 ± 0.130
	lion_cosine	49.993 ± 0.013	49.989 ± 0.012	0.767 ± 0.127	0.805 ± 0.076
Walker2d-v4	adamw	1.86 ± 0.81	2.03 ± 0.82	0.480 ± 0.147	0.485 ± 0.133
	lion_raw	1.47 ± 0.62	1.58 ± 0.58	0.369 ± 0.129	0.377 ± 0.114
	lion_cosine	2.41 ± 1.73	3.72 ± 2.10	0.249 ± 0.122	0.243 ± 0.116
	lion_warmup_cosine	1.66 ± 0.83	1.85 ± 0.89	0.331 ± 0.127	0.342 ± 0.118
Hopper-v4	adamw	2.13 ± 1.32	4.38 ± 2.61	0.377 ± 0.179	0.400 ± 0.161
	lion_raw	2.14 ± 1.42	2.70 ± 1.74	0.387 ± 0.188	0.387 ± 0.181

	lion_cosine	1.58 ± 0.81	2.46 ± 1.65	0.281 ± 0.231	0.313 ± 0.220
	lion_warmup_cosine	2.19 ± 1.45	2.31 ± 1.37	0.457 ± 0.184	0.470 ± 0.196
Ant-v4	adamw	25.37 ± 12.72	25.50 ± 13.79	0.564 ± 0.107	0.566 ± 0.101
	lion_raw	34.47 ± 13.73	30.44 ± 13.15	0.461 ± 0.108	0.437 ± 0.127
	lion_cosine	44.12 ± 8.29	43.72 ± 8.57	0.368 ± 0.062	0.368 ± 0.058
	lion_warmup_cosine	39.83 ± 10.98	38.24 ± 12.34	0.449 ± 0.077	0.455 ± 0.087

Recovery time in the HalfCheetah-v4 environment reaches approximately 50 s (the full time capacity of the fine-tuning window itself) for every model, with a standard deviation of less than 0.02 s — this metric does not constitute significant evidence of convergence, but is simply a ceiling effect resulting from the absence of a fall early-termination condition in this environment; we report it for completeness of presentation without building any conclusions upon it. In the three environments where recovery time constitutes an informative metric, two main patterns emerge:

- **Post-Fine-Tune Recovery Prolongation**: across the Walker2d-v4, Hopper-v4, and Ant-v4 environments, recovery time lengthens following fine-tuning for the vast majority of arm/environment combinations rather than shrinking — this is most clearly manifested in the lion_cosine model on the Walker2d-v4 environment (from 2.41 s to 3.72 s, an increase of 54%) and the adamw model on the Hopper-v4 environment (from 2.13 s to 4.38 s, more than doubling). This is consistent with the fact that fine-tuning under repeated perturbations shifts the policy toward a different operating point, not necessarily one that settles faster or is more damped.
- **Relative Chattering Stability**: chattering before and after fine-tuning remains within 0.01–0.03 of its pre-fine-tuning value for most arm/environment pairs, with lion_raw on the Ant-v4 environment constituting the most notable exception (from 0.461 to 0.437, a decrease of 5.2%).

Statistically Significant Shifts Under Perturbation

Across the four primary environments, Welch/Holm-corrected comparisons of each model's metrics in its clean conditions against each perturbation condition show a clear asymmetry directly linked to the mechanism at work: **sensor noise produces a confirmed statistically significant increase in chattering for the vast majority of tested models, while added mass fails to produce this effect almost entirely.**

- **Substantial Increases in Chattering Under Sensor Noise**: Hopper-v4 — lion_cosine records the largest effect size ($d = -3.50$), followed by lion_raw ($d = -2.74$), then lion_warmup_cosine ($d = -1.89$), and adamw ($d = -1.53$). Walker2d-v4 — lion_warmup_cosine records the largest effect size ($d = -3.42$), followed by lion_cosine ($d = -2.62$), then lion_raw ($d = -1.39$), and adamw_tuned ($d = -1.08$). Ant-v4 — adamw_tuned shows a sharp shift ($d = -2.25$).
- **Sharp Asymmetry Between Perturbation Types**: of the 20 (model/environment) combinations tested for chattering shift under sensor noise, 9 cases achieved Holm statistical significance, all in a single direction (increased chattering). In contrast, of the 40 (model/environment/mass level) combinations tested under added mass, only a single case reached the level of statistical significance (Ant-v4 with lion_warmup_cosine at mass +50%, at $d = -1.16$).

By contrast, return declines sharply under both types of perturbation for the majority of models across the four environments (30 out of 60 return comparisons reached significance under Holm correction), with effect sizes peaking at $d = 5.40$ (Walker2d-v4, lion_cosine, at mass +50%).

This asymmetry has a direct analytical reading in classical control, and specifically is consistent with **Sliding-Mode Control (SMC)** frameworks [3]: measurement noise near the switching surface is the typical trigger for the chattering phenomenon, and is distinguished as an addition to the discretization-driven chattering attributed to Lion's update rule (see the Method section above). The switching decision in the relay controller is directly disturbed by noise injected into the signal on which it bases its switching, whereas the added-mass perturbation merely modifies the plant dynamics without corrupting the measurements with which the policy interacts directly. The fact that sensor noise consistently escalates chattering in this data, while the mass perturbation lacks this consistency, aligns entirely with this interpretation, while acknowledging that the perturbation battery was not specifically designed to isolate this mechanism; we present this as a post-hoc descriptive pattern rather than as a test explicitly designed for that mechanism.

Humanoid-v4: Extended-Horizon Trajectories and Perturbation Behavior

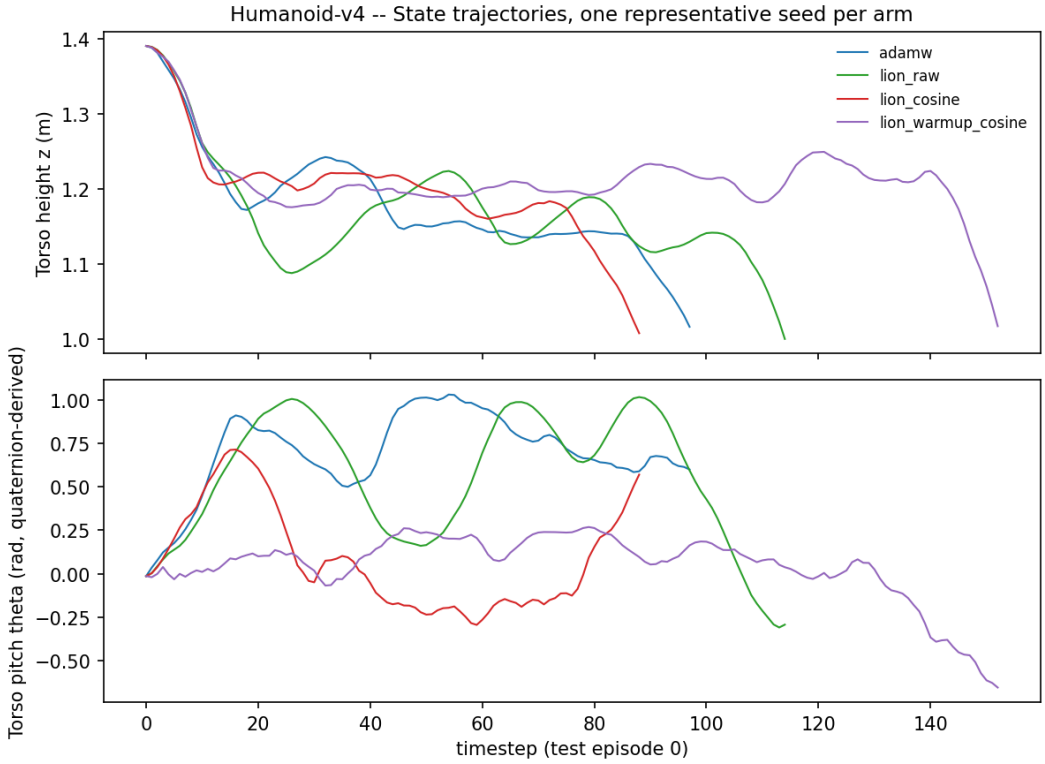

Figure 1. Humanoid-v4, one representative clean-condition evaluation episode per arm: torso height z (top) and torso pitch θ, quaternion-derived (bottom), against timestep. Each trace terminates at that seed's fall. lion_warmup_cosine (purple) survives longest in this representative episode; lion_cosine (red) falls soonest.

adamw_tuned was excluded from this extension due to the absence of evaluation data on return, chattering, or control metrics for it in the Humanoid-v4 environment within the experimental records; recording was limited to training wall-clock time (840.00 minutes versus 510.00 minutes for adamw), consistent with the systematic exclusion of this non-equivalent-condition model from the Holm-corrected inferential family regardless of the environment's nature. **Every policy evaluated on Humanoid-v4, under every arm and condition including clean conditions, falls before reaching the evaluation horizon (a 100% fall rate)**, unlike the four

primary environments where fall behavior depended on the nature of the arm and environment (Table 2); mean episode length, not fall rate, becomes the only available indicator of the policy's ability to defer failure, such that every metric below describes the dynamics occurring on the path toward that fall.

Figure 1 shows representative single-seed trajectories. All four curves start from the same initial torso height ($z \approx 1.39$ m) and descend toward the fall threshold, yet they move according to clearly divergent average dynamics: both adamw and lion_raw record an early recovery bump followed by a second drop; lion_cosine displays a similar early recovery but with a faster and earlier-synchronized final drop, consistent with it having the shortest survival duration among the three Lion models in this representative episode; while lion_warmup_cosine displays the most sustained near-plateau behavior (height oscillation within a relatively narrow range of 1.18–1.25 m from roughly timestep 40 onward) before it finally collapses at approximately step 150, exceeding by a wide margin the point at which the other three models ended. The torso pitch trajectory displays a consistent pattern: adamw and lion_raw show pitch oscillations of large amplitude (approaching ±1 rad) throughout the episode, while lion_cosine records a pitch trajectory of smaller amplitude though still oscillatory, and lion_warmup_cosine's pitch trajectory becomes clearly the most damped for the majority of the episode until a late sharp deviation precedes its fall.

Table 5. Humanoid-v4 (extended horizon), clean condition: joint-level diagnostics, $n = 15$ seeds, matched-condition arms only. Mean return (Welch/Holm-corrected, $n = 15$): adamw 484.2, lion_raw 445.0, lion_cosine 498.6, lion_warmup_cosine 522.7; lion_cosine and lion_warmup_cosine both significantly exceed lion_raw ($d = -1.09$ and $d = -1.28$, both p_holm < 0.05) and are statistically tied with each other ($d = -0.16$, ns); adamw is not significantly different from any Lion variant ($|d| \leq 0.52$, all ns).

Arm	Mean contact force (N)	Peak contact force (N)	Spike count/episode	Torque chattering	Saturation fraction
adamw	122.9	1319.0	4.21	0.179	0.924
lion_raw	115.1	1362.5	3.77	0.163	0.765
lion_cosine	120.1	1089.2	4.11	0.242	0.730
lion_warmup_cosine	118.7	1183.5	4.88	0.194	0.763

adamw drives the actuators at or beyond the torque limit threshold in the largest share of (step, joint) cases, at 0.924, a proportion that clearly exceeds all three Lion variants (0.730–0.765) — the same direction observed in the Hopper-v4 environment among the primary environments (Table 2), although the ranking there is contingent on the type of environment rather than being an absolute property of the optimizer. lion_cosine's chattering, with its value of 0.242, is significantly higher than both adamw ($d = -1.08$, p_holm = 0.034) and lion_raw ($d = -1.37$, p_holm = 0.006), consistent with the proposition that the model with the most disturbed and oscillatory representative trajectory in Figure 1 also carries the highest measured chattering value.

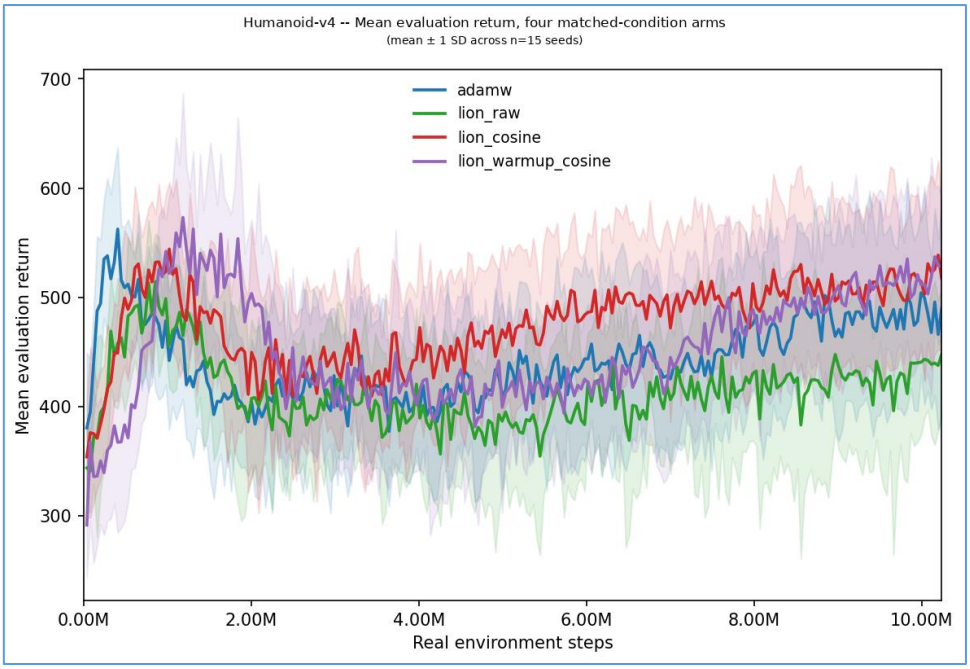

Figure 2. Humanoid-v4, mean evaluation return over the full 5000-iteration training horizon (≈10.24M environment steps), matched-condition arms, mean ± 1 SD across n = 15 seeds.

Figure 2 shows that this ranking is not merely a product of evaluating the final checkpoint: the lion_raw curve lies below the other three models for most of the training period, consistent with its significantly lowest final return (Table 5); while adamw, lion_cosine, and lion_warmup_cosine overlap substantially due to wide seed variance, consistent with the pairwise comparisons among them in Table 5 not reaching the level of Holm significance.

Table 6. Humanoid-v4: mean return under perturbation as a percentage of each arm's own clean-condition mean return, *n* = 15 seeds per cell. This table is exploratory and descriptive only, and no Welch/Holm significance testing was run on the underlying Humanoid-v4 perturbation returns.

Arm	Mass +20%	Mass +50%	Sensor noise
adamw	**54.1%**	**41.5%**	**72.3%**
lion_raw	50.3%	43.7%	62.8%
lion_cosine	38.9%	34.9%	57.8%
lion_warmup_cosine	44.0%	38.1%	52.7%

Table 6, at the level of arms within a single environment, expresses the same pattern documented across environments above, namely that clean-conditions performance does not constitute a predictive indicator of robustness: adamw retains the largest fraction of its own clean return under all three perturbations, whereas lion_cosine — statistically tied with lion_warmup_cosine as the most efficient model in clean conditions for the Humanoid-v4 environment, and numerically superior to adamw and lion_raw (Table 5) — records the lowest retained fraction under both mass perturbations. We caution explicitly against interpreting settling time and overshoot percentage in this environment as classical control-quality indicators: settling time (on the torso-height trajectory) decreases monotonically from clean conditions to mass +20% then to mass +50% for every model, tracking the corresponding shrinkage in mean episode length closely enough that the apparent "faster settling" phenomenon is at least partly an artifact resulting from the episode's earlier termination rather

than evidence of higher damping; for this reason we omit these two metrics from the comparative tables above and confine them to reporting within the underlying data.

Chattering as a Sliding-Mode Phenomenon, With a Noise-Specific Signature

The Method section above provides an analytical justification of chattering as the parameter-space equivalent of the sliding-mode-control phenomenon documented by [3]. The cross-environment pattern above comes to crown and refine this interpretation: chattering increases substantially under sensor noise for roughly half of the tested (model/environment) combinations, while this effect almost entirely vanishes under added mass — a distinction that is structurally consistent with the distinction adopted in Sliding-Mode Control theory between chattering caused by corruption of measurements at the switching surface and that caused by discretization or unmodeled dynamics. In the Humanoid-v4 environment, the model with the apparently most damped representative trajectory (lion_warmup_cosine) holds the lowest chattering value in clean conditions among the scheduled Lion variants, alongside the longest mean episode length in clean conditions; conversely, the model with the most oscillatory pitch trajectory within the Lion family, namely lion_cosine, records the highest chattering value and the shortest survival duration under both mass perturbations there. Together, these patterns converge to confirm chattering's function as an early indicator of control-quality decline in this type of task — even though they do not on their own establish causal evidence — with the specific nature of the perturbation emerging as important in shaping the control-theory-grounded mechanism.

Clean-Condition Performance Does Not Predict Perturbation Robustness

The clearest and most cross-cutting finding in this research is that a model's ranking in clean conditions is a weak predictor of its ranking in robustness against perturbations, and that this dissociation manifests within a single environment, not only across differing environments:

- **Walker2d-v4**: lion_cosine is among the top-performing models in clean conditions, yet it records the lowest recovery profile.
- **Ant-v4**: lion_raw represents the weakest Lion model in clean conditions, yet it records the highest recovery profile in the entire table.
- **Humanoid-v4**: lion_cosine again stands out among the top-performing models in clean conditions, yet it retains the lowest fraction of its own return under both mass perturbations.

These findings constitute a more precise formulation of the observation evident at the environment level above (Table 3), namely that passive mechanical stability and robustness against perturbations are complementary but conceptually distinct properties; here, the same dissociation is revealed between models sharing the same environment and, in two out of three cases, the same optimizer family. We recommend that practitioners weighing optimizer configurations and learning-rate schedules for systems in which physical perturbations during operation constitute a decisive priority should conduct direct validation under the target perturbation environment rather than direct inference from standard models' performance in clean conditions, however strong that performance may be.

Practical Implications

Two practical implications follow from this. First, if robustness against sensor/observation noise specifically constitutes a major concern during field operation, then the chattering metric tracked in this paper is a relevant leading indicator, apparently noise-specific, and deserves close monitoring during training rather than being confined to final evaluation alone. Second, recovery time tends to lengthen following fine-tuning rather than shrink relative to pre-fine-tuning behavior, suggesting that the short re-adaptation window under repeated perturbations does not directly restore or improve settling behavior even when it restores return; return-based notions of recovery and settling-time-based ones may conflict, and any field operating system

that limits itself to monitoring return risks overlooking a control-quality decline that return-only dashboards are unable to reveal.

Limitations

The scope limitations disclosed at the points where they arise throughout this paper are consolidated here to avoid repetition.

- **Humanoid-v4**: all evaluated policies fall before reaching the episode horizon under every arm and condition, and hence its metrics describe the dynamics occurring on the path toward falling rather than a mature controller; furthermore, its perturbation-return table (Table 6) is exploratory in nature and has not undergone statistical significance testing.
- **Recovery time on HalfCheetah-v4**: this was censored at the full time capacity of the fine-tuning window itself for every model (Table 4), and is not a genuine convergence metric in this environment.
- **Settling time/overshoot on Humanoid-v4**: these were confounded by episode-length truncation, and were accordingly omitted from the comparative tables.
- **Modeling scope**: this study is limited to evaluating a single reinforcement-learning algorithm (PPO) and a fixed perturbation battery (added mass and sensor noise only; without including actuator delay or friction perturbation).
- **Trajectory sampling**: the trajectory depiction in Figure 1 is limited to one representative seed per model rather than showing the full distribution across $n = 15$ seeds.
- **Absence of field validation (sim-to-real)**: the research lacks any validation on real physical hardware (sim-to-real); all results remain based exclusively on simulation.

These scope limitations do not weaken the robustness of the paper's central claims, but they should be carefully weighed by the reader when assessing the extent to which the findings generalize beyond the specific algorithm, environments, and perturbation battery examined in this study.

Conclusion

This paper examined the control-theory-based and trajectory-level behavior of the Lion and AdamW optimizers in the PPO algorithm, using per-environment diagnostics and an integrated perturbation protocol across all five environments. The paper arrives at three main findings:

- Chattering increases substantially under sensor noise for roughly half of the tested (model/environment) combinations, while this effect nearly vanishes under added mass, a pattern that has a direct analytical reading in classical sliding-mode control theory.
- Post-fine-tuning return recovery varies enormously depending on the arm within a single environment: the lion_cosine model, among the highest-performing Walker2d-v4 models in clean conditions, records the lowest recovery profile of any arm in this study on that environment, while lion_raw, the weakest Lion model in clean conditions on the Ant-v4 environment, holds the best recovery profile.
- The exploratory extension to the Humanoid-v4 environment — where all policies fall in every evaluation episode — reveals that AdamW retains a higher fraction of its own clean return under all three perturbations compared with any Lion variant, **which also reverses the clean-conditions ranking there** (AdamW is not the highest-performing model in clean conditions on Humanoid-v4, yet it becomes the most robust against perturbations).

Together, these findings show, at a degree of precision exceeding what any aggregate environment-level summary could capture, that robustness against perturbations cannot be

reliably predicted based on clean-conditions performance, and that direct validation is required for any field applications in which physical perturbations constitute a primary design concern. As a way forward, we recommend that practitioners deploying PPO-trained policies in settings where physical perturbations are expected treat optimizer and schedule choice as a robustness-relevant design decision to be validated directly, rather than one to be inferred from clean-condition benchmarks alone.

References

1. X. Chen, C. Liang, D. Huang, E. Real, K. Wang, Y. Liu, H. Pham, X. Dong, T. Luong, C.-J. Hsieh, Y. Lu, and Q. V. Le, "Symbolic discovery of optimization algorithms," in *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.
2. J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," in *Proc. 35th Int. Conf. Machine Learning (ICML)*, PMLR vol. 80, pp. 560–569, 2018.
3. V. I. Utkin, "Variable structure systems with sliding modes," *IEEE Transactions on Automatic Control*, vol. 22, no. 2, pp. 212–222, 1977.
4. T. Dettmers, M. Lewis, S. Shleifer, and L. Zettlemoyer, "8-bit optimizers via block-wise quantization," in *International Conference on Learning Representations (ICLR)*, 2022.
5. J. Zhao, Z. Zhang, B. Chen, Z. Wang, A. Anandkumar, and Y. Tian, "GaLore: Memory-efficient LLM training by gradient low-rank projection," in *Proc. 41st Int. Conf. Machine Learning (ICML)*, 2024.
6. Y. Dong, H. Li, and Z. Lin, "Convergence rate analysis of LION," *arXiv preprint arXiv:2411.07724*, 2024.
7. K. Liang, L. Chen, B. Liu, and Q. Liu, "Cautious optimizers: Improving training with one line of code," *arXiv preprint arXiv:2411.16085*, 2024.
8. N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. 5th Conf. Robot Learning (CoRL)*, PMLR vol. 164, pp. 91–100, 2022.
9. A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," in *Robotics: Science and Systems (RSS) XVII*, 2021.
10. P. Wu, W. Xie, J. Cao, H. Lai, and W. Zhang, "LoopSR: Looping sim-and-real for lifelong policy adaptation of legged robots," *arXiv preprint arXiv:2409.17992*, 2024.
11. T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.
12. T. He, J. Gao, W. Xiao, Y. Zhang, Z. Wang, J. Wang, Z. Luo, G. He, N. Sobanbabu, C. Pan, Z. Yi, G. Qu, K. Kitani, L. Fan, Y. Zhu, J. Hodgins, C. Liu, and G. Shi, "ASAP: Aligning simulation and real-world physics for learning agile humanoid whole-body skills," in *Robotics: Science and Systems (RSS)*, 2025.
13. I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations (ICLR)*, 2019.
14. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.
15. F. H. Clarke, *Optimization and Nonsmooth Analysis*. Wiley, 1983.
16. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
17. E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *2012 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pp. 5026–5033, 2012.

18. M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. De Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis, "Gymnasium: A standard interface for reinforcement learning environments," *arXiv preprint arXiv:2407.17032*, 2024.
19. A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-Baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.
20. L. Engstrom, A. Ilyas, S. Santurkar, D. Tsipras, F. Janoos, L. Rudolph, and A. Madry, "Implementation matters in deep RL: A case study on PPO and TRPO," in *International Conference on Learning Representations (ICLR)*, 2020.
21. M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, S. Gelly, and O. Bachem, "What matters for on-policy deep actor-critic methods? A large-scale study," in *International Conference on Learning Representations (ICLR)*, 2021.
22. J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *International Conference on Learning Representations (ICLR)*, 2016.
23. S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.
24. J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*, 2nd ed. Hillsdale, NJ: Lawrence Erlbaum Associates, 1988.
25. C. Colas, O. Sigaud, and P.-Y. Oudeyer, "How many random seeds? Statistical power analysis in deep reinforcement learning experiments," *arXiv preprint arXiv:1806.08295*, 2018.

Appendix: adamw_tuned Per-Environment Hyperparameters

Table 7. adamw_tuned configuration by environment (externally tuned per RL Zoo; the only hyperparameters that vary by environment).

Hyperparameter	HalfCheetah-v4	Hopper-v4	Walker2d-v4	Ant-v4
Optimizer	AdamW	AdamW	AdamW	AdamW
Initial learning rate η_0	2.0633×10^{-5}	9.80828×10^{-5}	5.05041×10^{-5}	1.90609×10^{-5}
LR schedule	constant	constant	constant	constant
PPO epochs/update	20	5	20	10
Minibatch size	64	32	32	32
Rollout n_steps	512	512	512	512
β_1	0.9	0.9	0.9	0.9
β_2	0.999	0.999	0.999	0.999
Weight decay λ	0.0	0.0	0.0	0.0
Clip range ε	0.1	0.2	0.1	0.1
Discount γ	0.99	0.999	0.99	0.98
GAE λ	0.95	0.99	0.95	0.80
max_grad_norm	1.0	0.7	1.0	0.6
vf_coef	0.871923	0.835671	0.871923	0.677239
ent_coef	5.85045×10^{-4}	2.29519×10^{-3}	5.85045×10^{-4}	4.9646×10^{-7}
Hidden layer width	256/256	256/256	64/64	not specified
Random seeds	15	15	15	15
RL Zoo adoption status	officially adopted	officially adopted	officially adopted	commented-out block*

Network architecture (hidden-layer width) for the adamw_tuned column varies by environment per the official policy_kwargs in RL Zoo: 256×256 for HalfCheetah-v4 and Hopper-v4; Walker2d-v4's tuned block contains no custom policy_kwargs, so the SB3 MlpPolicy default is used. *The Ant-v4 (tuned) block is fully commented out in the official ppo.yml, and Ant-v4 is not among the 10 officially tuned MuJoCo environments in the repository. Values in the Ant-v4 column above are taken from that commented-out block and should be read as a practitioner-published reference rather than a validated, officially adopted configuration. adamw, lion_raw, lion_cosine, and lion_warmup_cosine architectures were fixed at 128×128 across all four environments as a deliberate methodological choice, to isolate the effect of the optimizer from the effect of network capacity, after a pilot comparison showed no material sensitivity to this choice.

Compliance with ethical standards

Disclosure of conflict of interest

The authors declare that they have no conflict of interest.